**FY2003 UNIVERSITY SOFTWARE INITIATIVE PROPOSAL**
**FOR THE**
**NASA SOFTWARE IV&V FACILITY**

Initiative Title: **Sensitivity of Software Reliability to Operational Profile Errors: Architecture-Based Approach**

Initiative ID: **Project 10005549, Award 1002193R**

*May 2003 deliverable:*

Report on

# Uncertainty analysis of software operational profile based on perturbation theory


**Katerina Goseva–Popstojanova**

**Sunil Kumar Kamavaram**

*LANE Department of Computer Science and Electrical Engineering*
*West Virginia University*

## Report Summary

In this report we address the use of perturbation theory for uncertainty analysis of software operation profile. First, we briefly describe the perturbation theory and then we explain how we relate this theory to our research work. The perturbation theory as a method for uncertainty analysis is applied and validated on three case studies: the software developed for the European Space Agency, NASA's Hub Control System (HCS) from the International Space Station (ISS), and an e-commerce application. Finally, we present the future directions of our research on uncertainty analysis in software reliability.

## Table of Contents

# 1. Introduction

Software reliability is defined as the probability that software product will work without failure in a specified environment for a specified exposure period. The exposure period can be execution time or software runs. The environment usually is characterized by a set of input states along with their probabilities of occurrence. This probability distribution over the input space that represents the frequencies of occurrence of possible input states in the operation of software application is known as operational profile. Thus, the predictive quality of software reliability models is affected by the ability to estimate the correct operational profile. However, building an operational profile is not an easy task, especially for a new product. Therefore, it is of critical importance to study the sensitivity of the software reliability to the errors in the operational profile, particularly when reliability estimates with high accuracy are required.

In general, the estimation of a trustworthy operational profile is difficult because it requires anticipating the field usage of the software and a priori knowledge about the application and system environments. A typical example would be a flight control system of a spacecraft in which very critical software components are activated by physical events whose frequencies during the field usage are totally unknown. Further, in process control applications various software components are activated by complex sequences of events whose frequencies can hardly be estimated a priori. In other cases, a single operational profile is not sufficient to describe the use of the product by different users. Because the effort required to derive an operational profile for each group of users is usually extremely high, the usual solution is to adopt an approximate operational profile that represents a rough average of the operational profiles of the different users. In addition to the above difficulties, problems could arise due to the changes of the operational profile during the development and field usage of the software. Thus, software systems evolve because functions are added or modified. As a consequence, the way in which the software is used also evolves, and the operational profile changes. This, of course, will invalidate any existing estimates of the operational profile.

These reasons can easily lead to erroneous estimates of the operational profile which will directly affect the reliability estimate. Studying the variations of the reliability estimate due to the inaccuracy in the operational profile is especially important for NASA domain software which is designed to deal with events whose frequencies are difficult or impossible to predict accurately. The objective to this project is to develop an architecture-based methodology for computing the sensitivity of software reliability to the operational profile errors.

In the first year of this project (FY02) we have developed a methodology for uncertainty analysis of architecture-based software reliability models suitable for large complex component-based applications and applicable throughout the software life cycle [Goseva02a], [Goseva02b]. In order to estimate the system reliability using architecture-based model we need to know the software architecture (structure of component interactions), software usage described by the operational profile (relative frequencies of component interactions determined by transition probabilities), and software failure behavior (component reliabilities or failure rates) [Goseva01a], [Goseva01b]. Our methodology considers different approaches for building software architecture (intended approach and informed approach) and estimating component reliabilities (growth models, non-failed executions, and fault injection) as shown in Figure 1. This methodology for uncertainty analysis can be applied to any architecture-based software reliability model that has a close form solution for the system reliability. It addresses the parameter uncertainty problem and enables us to study how the uncertainty of parameters propagates in the system reliability. Within this methodology we are considering several different methods for uncertainty analysis (see Figure 2) [Goseva02b]. So far, we have used entropy [Kamavaram02], [Goseva02f], methods of moments [Goseva02d], [Goseva02e] and Monte Carlo simulation [Goseva02e]. We have applied and validated our methodology on two case studies [Goseva02b]: software developed for the European Space Agency and NASA's Hub Control System (HCS) from the International Space Station (ISS). A detailed description of the application of our methodology on these case studies can be found in [Goseva02b].

The choice of the method for uncertainty analysis will depend on criteria such as data requirements, reliability measures derived, accuracy of the solutions, and scalability with respect to the number of components. Detailed comparison of the entropy, method of moments, and Monte Carlo simulation is presented in [Goseva02c].

In this report we focus on perturbation theory as a method for uncertainty analysis of the software operational profile. First, we briefly describe the perturbation theory and then apply it on three case studies: the software developed for the European Space Agency, NASA's Hub Control System (HCS) from the International Space Station (ISS), and an e-commerce application.

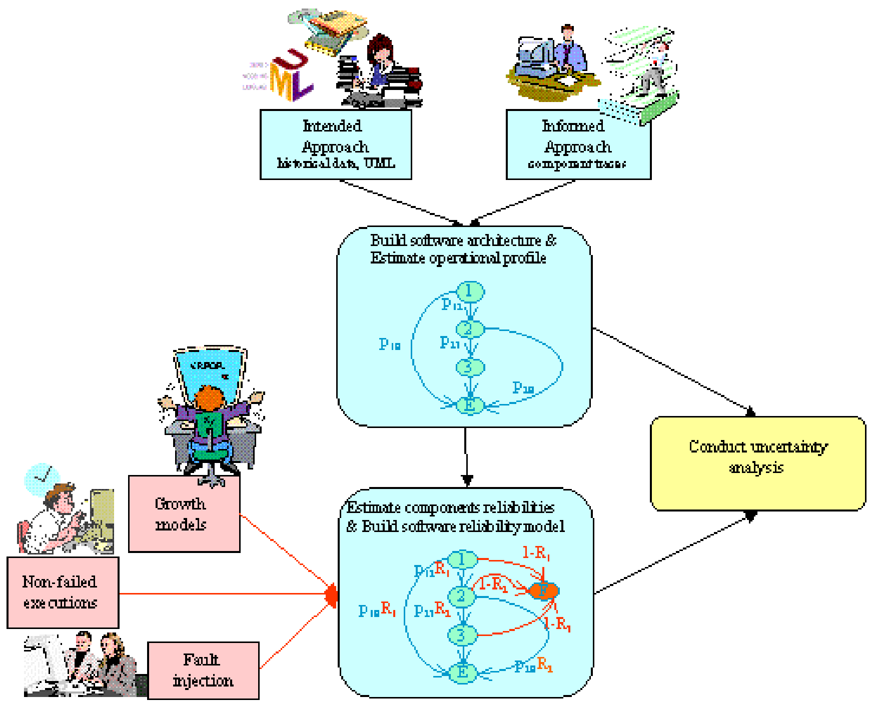


**Figure 1. Methodology for uncertainty analysis of software reliability**

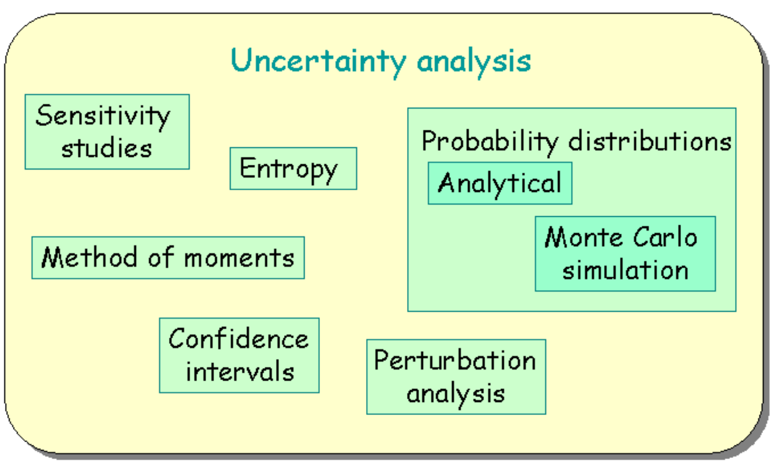


**Figure 2. Methods for uncertainty analysis in software reliability**

## 2. Sensitivity analysis based on perturbation theory

Perturbation theory provides mathematical means to study how the stationary distribution of a Markov chain containing an irreducible set of states changes as the transition probabilities of the chain vary. In our research we define the operational profile as a discrete time Markov chain with transition probability matrix P. Using perturbation theory we can assess the sensitivity of stationary probabilities $\pi = [\pi_i]$ to perturbations in the operational profile (i.e, transition probability matrix P). Since $\pi_i$ can be interpreted as the expected execution rate of component *i* in the long run, it represents a measure of component usage which can be used to identify critical components.

Let P be the transition probability matrix of a finite irreducible Markov Chain and $\pi = [\pi_i]$ be the stationary probability vector. The stationary distribution vector of P is the positive vector $\pi^T = (\pi_1, \pi_2, \ldots\ldots, \pi_n)$ satisfying

$$\pi^T P = \pi^T, \qquad \sum_{j=1}^{n} \pi_j = 1. \tag{1}$$

Suppose P is perturbed to a matrix $\tilde{P}$, which is the transition probability matrix of an *n* state finite irreducible, homogenous Markov chain. Denoting by $\tilde{\pi}$ the stationary probability vector of the perturbed matrix $\tilde{P}$, the aim is to assess the sensitivity of the stationary distribution vector in terms of the change $E \equiv P - \tilde{P}$ in the transition probability matrix. Sensitivity results concerning absolute perturbations have been phrased in terms of bounds given by the equation [Ipsen94]

$$\left\|\pi^T - \tilde{\pi}^T\right\| \le k\|E\| \tag{2}$$

where $\|E\|$ is the norm of the perturb matrix and *k* is a condition number used as measure of sensitivity. In the literature survey we came across eight existing perturbation bound, that is, eight different condition numbers $k_1, \ldots, k_8$. Most of the condition numbers are expressed in terms of either the fundamental matrix $Z \equiv (A + e\pi^T)^{-1}$ of the underlying Markov chain or the group inverse of A≡I-P [Meyer94], [Schweitzer68]. While several condition numbers provide good numerical measure of the maximal extent to which the magnitude of the perturbation can be amplified, some condition numbers suffer from certain shortcomings and are not satisfying for two reasons. First, irreducible chains exist for which the bounds are not tight, so the condition number $k$ may seriously overestimate the sensitivity to perturbations. Second, the bounds generally provide very little information about the relative error in individual stationary probabilities. Moreover, it is theoretically possible to compute the condition numbers $k$ but it is usually expensive to do so relative to computation of the stationary distribution vector itself [Meyer94], [Cho00], [Cho01].

The condition number $k_{8,}$ expressed in terms of the mean first passage times in the Markov chain, is the smallest condition number, that is, it provides the tightest bound on the stationary probability. Further, viewing sensitivity in terms of mean passage times can sometimes help practitioners decide whether or not to expect sensitivity by merely observing the structure of the chain without computing or estimating the condition numbers. For example, no stationary probability of the Markov chain consisting of a dominant central state with connections to and from all other states can be excessively sensitive to perturbations in P.

## 2.1. Sensitivity analysis in terms of mean first passage times

Let P and $\tilde{\mathrm{P}}$ be the transition probability matrices for the two irreducible $n$ state Markov chains with respective stationary probability vectors $\pi^T$ and $\tilde{\pi}^T$. Let $M_{ij}$ denote the mean first passage time from the $i^{\text{th}}$ state to the $j^{\text{th}}$ state and $M_{jj}$ denote the mean return time for the $j^{\text{th}}$ state in the unperturbed chain. The absolute change in the $j^{\text{th}}$ state stationary probability is given by equation [Cho00]

$$\left|\pi_j - \tilde{\pi}_j\right| \le \frac{1}{2} \cdot \frac{\max_{i \ne j} M_{ij}}{M_{jj}} \cdot \|E\|_\infty, \tag{3}$$

which is equivalent to saying that the relative change in $\pi_j$ is

$$\frac{\left|\pi_j - \tilde{\pi}_j\right|}{\pi_j} \le \frac{1}{2} \cdot \max_{i \ne j} M_{ij} \cdot \|E\|_\infty \ . \tag{4}$$

In [Hunter00], [Hunter02a], [Hunter02b] a standard procedure for computing the mean first passage times of a finite irreducible discrete time Markov chains has been developed. If the stationary probability vector has already been computed then using either fundamental matrix

$Z \equiv [I - P + \Pi]^{-1}$, where $\Pi = e\pi^T$ or Meyer's group inverse [Meyer94] $A^{\#} \equiv Z - \Pi$, we can compute the matrix M. The diagonal elements of the matrix M specify the mean return time of the states and non-diagonal elements specify the mean first passage time from state *i* to state *j*. If $A^{\#} = [I - P + e\pi^T]^{-1} - e\pi^T = [a_{ij}^{\#}]$ then matrix M is given by equation [Hunter02a]

$$M = [m_{ij}] = [I - A^{\#} + L A_d^{\#}]D \tag{5}$$

where $L = ee' = [1]$, $e$ is a column vector with all elements equal to *1, e'* is a row vector with all elements equal to *1*, and $D = (\Pi_d)^{-1}$.

## 3. Application of perturbation analysis on case studies

### 3.1. European Space Agency case study

The application from the European Space Agency [Goseva01b] provides language-oriented user interface, which allows the user to describe the configuration of an array of antennas. Its purpose is to prepare a data file in accordance with a predefined format and characteristics from a user, given the array antenna configuration described using the Array Definition Language. The program was developed in C language and consists of almost 10,000 lines of code. It has been extensively used after the last fault removal without failures. This gold version was used as an oracle in the experiment. A set of test cases was generated randomly accordingly to the known operational profile determined by interviewing the users of the program. Component traces obtained during the testing were used for building the software architecture and estimating transition probabilities. It follows that for the ESA case study we use an informed approach. Figure 3 presents the special case of our methodology for uncertainty analysis used for the European Space Agency case study. Details about this case study are given in [Goseva01b], [Goseva02b].

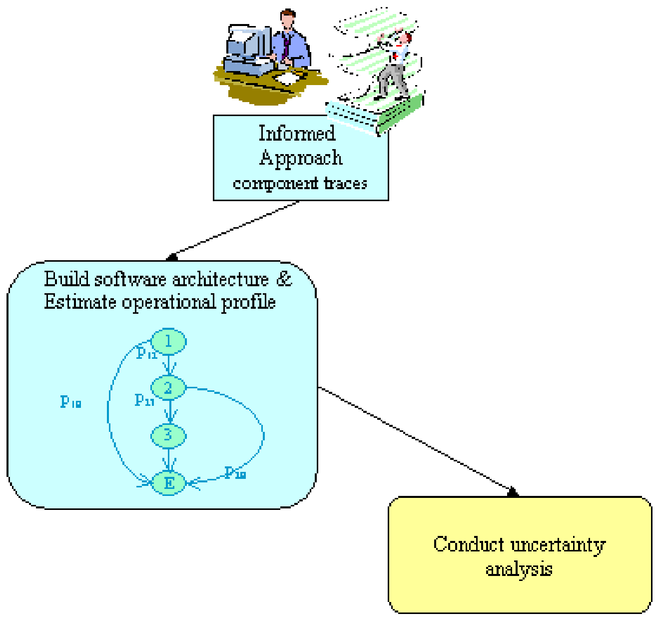


**Figure 3. Special case of the methodology used for the ESA case study**

DTMC that represents software architecture is shown in Figure 4. Components 1, 2, and 3 correspond to the Parser, Computational, and Formatting subsystems respectively. State E represents the completion of execution.

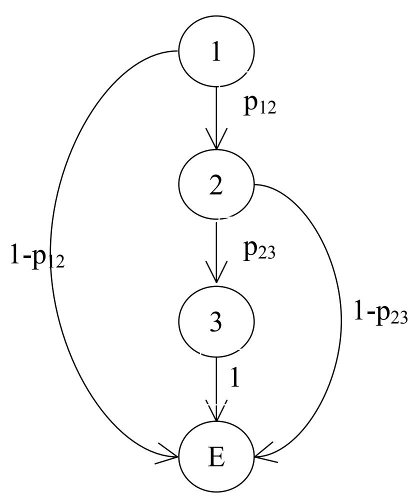


**Figure 4. Software architecture for the ESA case study**

In this experiment we look at the stability of the above Markov chain by perturbing the transition probability matrix, that is, the operational profile. Let us consider the transition probability matrix of operation profile A denoted by $P_A$ and let this matrix be perturbed to matrix $P_B$, which is the transition probability matrix of operational profile B of the same application. Matrices $P_A$, $P_B$, and the perturb matrix E are given below.

$$P_A = \begin{bmatrix} 0 & 0.5933 & 0 & 0.4067 \\ 0 & 0 & 0.7704 & 0.2296 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad E = \begin{bmatrix} 0 & 0.1431 & 0 & -0.1431 \\ 0 & 0 & -0.1018 & 0.1018 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad P_B = \begin{bmatrix} 0 & 0.7364 & 0 & 0.2636 \\ 0 & 0 & 0.6866 & 0.3134 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Next, we assess the sensitivity of the stationary probabilities (i.e., components execution rates) using the equations (3), (4) and (5). We can observe from Table 1 that the chain representing the operational profile A is absolutely stable since each stationary probability is insensitive to perturbations in $P_A$ in the absolute sense.

| **States** | **1** | **2** | **3** | **E** |
|---|---|---|---|---|
| **Stationary Probability** | 0.3278 | 0.1945 | 0.1498 | 0.3278 |
| **Absolute Change** | 0.12997 | 0.12166 | 0.1217 | 0.09619 |
| **Relative Change** | 0.39644 | 0.62549 | 0.8119 | 0.29341 |

**Table 1. Perturbation analysis of ESA case study**

We also consider a hypothetical example based on the European Space Agency application which has a loop back from component 2 to 1 [Goseva01b], [Goseva02b]. The software architecture for hypothetical example is shown in Figure 5.

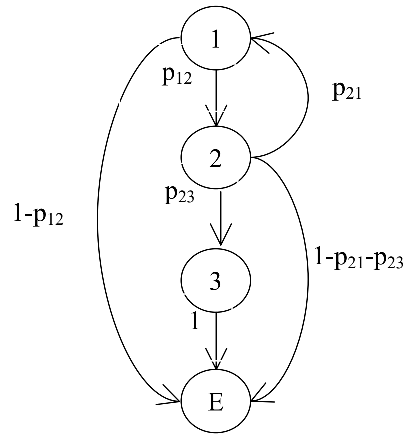


**Figure 5. Software architecture for the hypothetical example**

We assume that the transition probability matrix $P_C$ of the hypothetical example is perturbed by matrix E which results into a perturbed matrix $P_D$ as shown below.

$$P_C = \begin{bmatrix} 0 & 0.8 & 0 & 0.2 \\ 0.25 & 0 & 0.25 & 0.5 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad E = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & -0.5 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad P_D = \begin{bmatrix} 0 & 0.8 & 0 & 0.2 \\ 0.75 & 0 & 0.25 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

The sensitivity of the stationary probabilities is estimated using equations (3), (4) and (5). From the results given in Table 2 it can be seen that the operational profile is not stable in relative sense to the perturbations. In particular, the component 3, which has the smallest stationary probability, is the most sensitive in relative sense to the changes made to the transition probability matrix $P_C$.

| **States** | **1** | **2** | **3** | **E** |
|---|---|---|---|---|
| **Stationary Probability** | 0.35715 | 0.28571 | 0.07143 | 0.28571 |
| **Absolute Change** | 0.35714 | 0.5 | 0.46429 | 0.35714 |
| **Relative Change** | 1 | 1.75 | 6.5 | 1.25 |

**Table 2. Perturbation analysis of the hypothetical example**

### 3.2. Hub Control System case study

In this section we assess the sensitivity of NASA's HCS case study using perturbation theory. The case study is a Computer Software Configuration Item (CSCI) resident in the Hub Control Zone Multiplexers/Demultiplexers (HCZ MDMs) which are installed in the Node 3 Module of the ISS (International Space Station). For this case study we had available the UML use case diagram and sequence diagrams for each use case. Therefore, we are using the intended approach to build software architecture. We build DTMCs using UML sequence diagrams that present software components used for given scenario and the messages that are exchanged between these components. The expression used to estimate the transition probability from component *i* to component *j* is given by $p_{ij} = \frac{n_{ij}}{n_i}$, where $n_{ij}$ is the number of times messages are transmitted from component *i* to component *j* and $n_i$ is the total number of massages from component *i* to all other components that are present in the sequence diagram. Figure 6 presents the special case of our methodology used for the HCS case study.

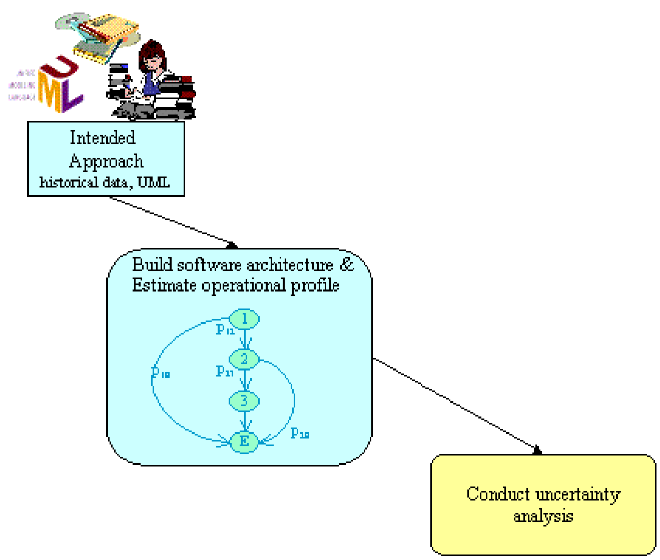


**Figure 6. Special case of the methodology used for HCS case study**

Figure 7 shows the main use case diagram and all the relationships among the use cases and the actors. Each use case is realized by at least one sequence diagram (i.e., scenario).

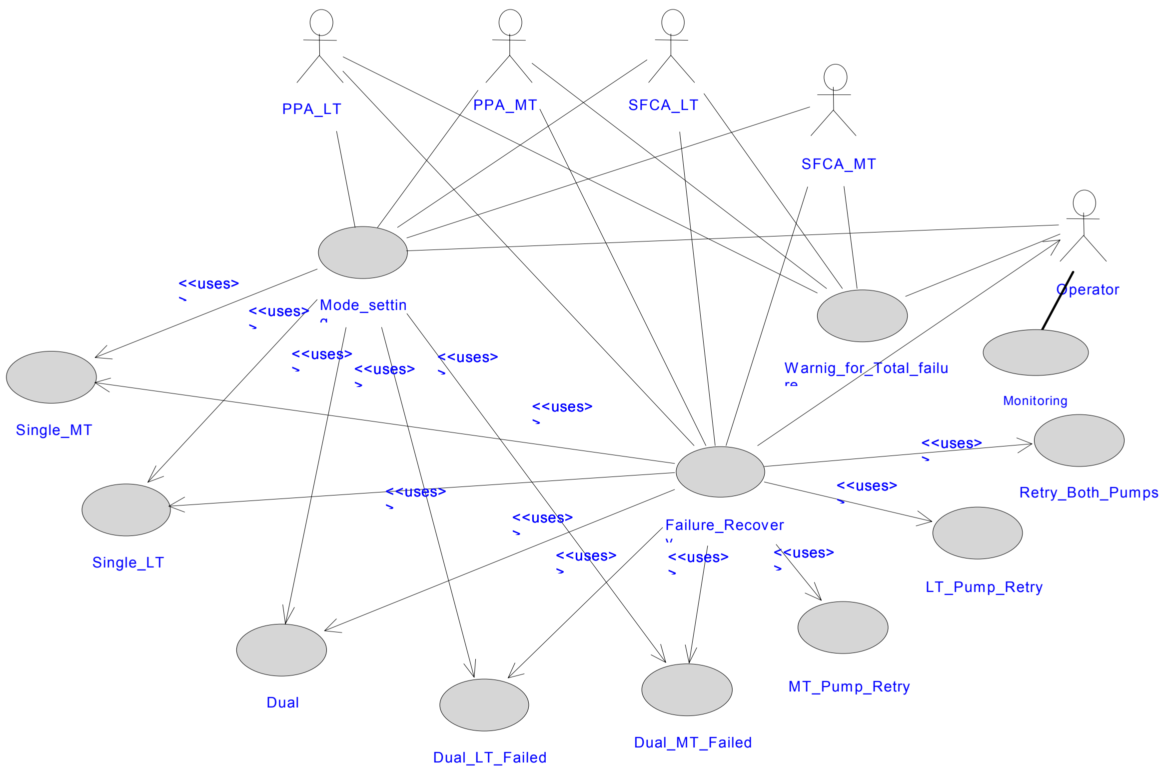


**Figure 7. Use case diagram of the HCS case study**

In this report we illustrate the application of the perturbation analysis on the *Both Pumps Retry* scenario. Figure 8 shows the sequence diagram of this scenario. Analyzing the sequence diagram of the *Both Pumps Retry* scenario given in Figure 8, we construct the DTMC that represents the software execution behavior as shown in Figure 9.

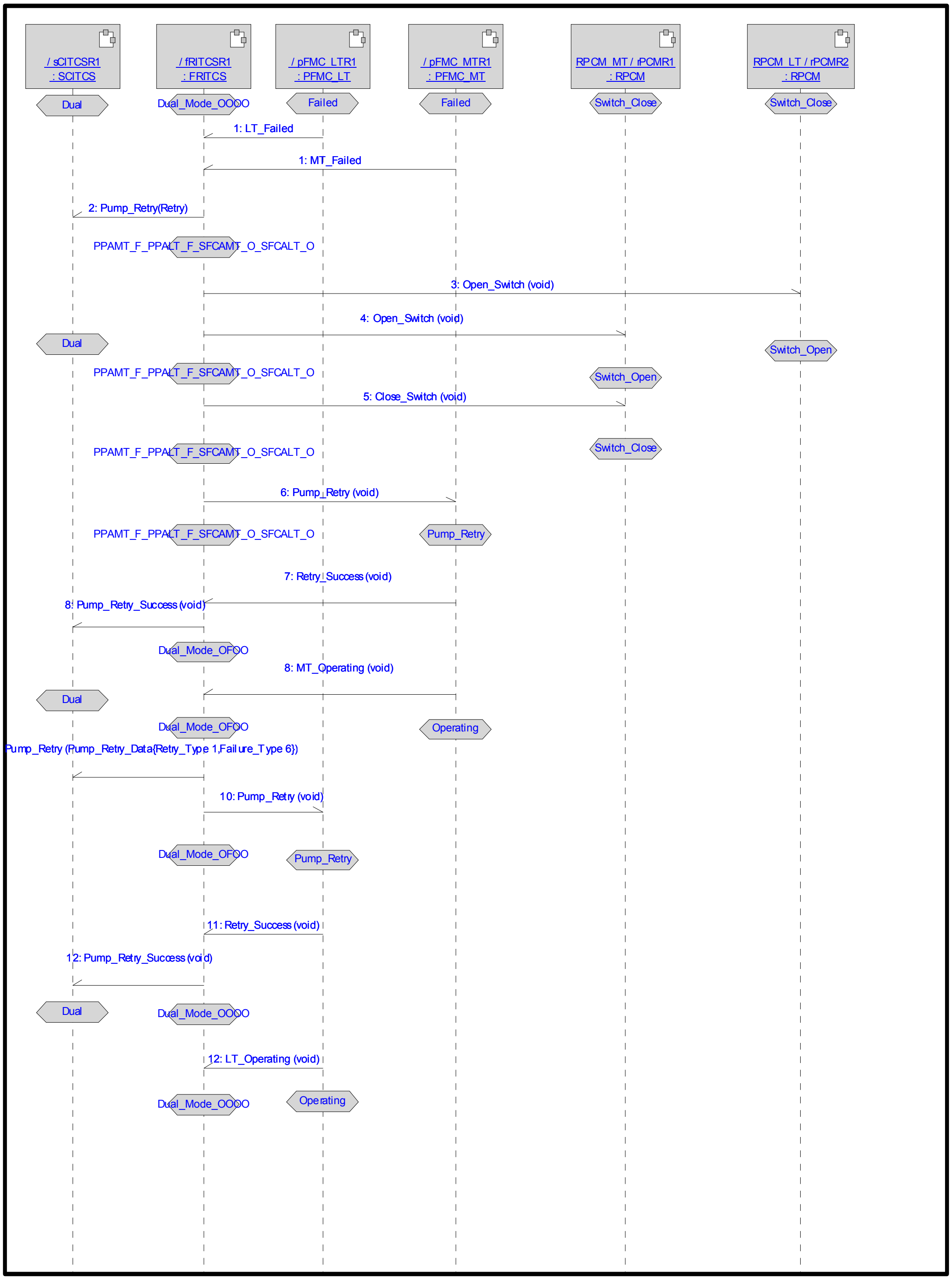


**Figure 8. Sequence diagram of the *Both Pumps Retry* scenario**

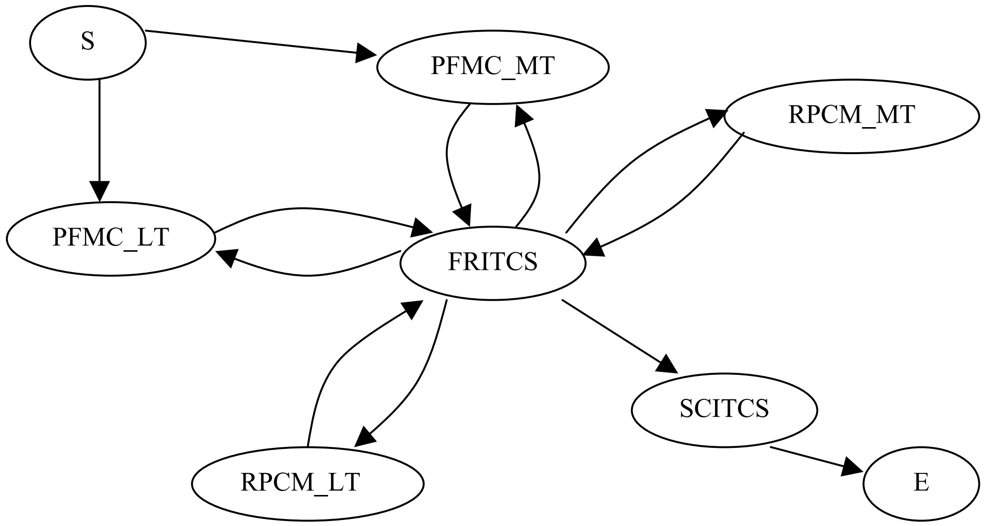


**Figure 9. DTMC for the *Both Pumps Retry* scenario**

The sensitivity of the operational profile of *Both Pumps Retry* scenario given in Figure 9 is analyzed by perturbing the transition probability matrix P. The resulting matrix $\tilde{\mathrm{P}}$ represents a different usage of the same scenario.

$$\mathrm{P} = \begin{array}{c} \text{S} \\ \text{PFMC\_LT} \\ \text{PFMC\_MT} \\ \text{FRITCS} \\ \text{RPCM\_LT} \\ \text{RPCM\_MT} \\ \text{SCITCS} \\ \text{E} \end{array} \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1/9 & 1/9 & 0 & 1/9 & 2/9 & 4/9 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathrm{E} = \begin{array}{c} \text{S} \\ \text{PFMC\_LT} \\ \text{PFMC\_MT} \\ \text{FRITCS} \\ \text{RPCM\_LT} \\ \text{RPCM\_RT} \\ \text{SCITCS} \\ \text{E} \end{array} \begin{bmatrix} 0 & -0.3 & 0.3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/9 & 1/9 & 0 & 1/9 & 0 & -1/3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\tilde{\mathrm{P}} = \begin{array}{c} \text{S} \\ \text{PFMC\_LT} \\ \text{PFMC\_RT} \\ \text{FRITCS} \\ \text{RPCM\_LT} \\ \text{RPCM\_MT} \\ \text{SCITCS} \\ \text{E} \end{array} \begin{bmatrix} 0 & 1/5 & 4/5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2/9 & 2/9 & 0 & 2/9 & 2/9 & 1/9 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The results of the perturbation analysis obtained using equations (3), (4), and (5) are given in Table 3. Note that the execution rates of components FRITCS and RPCM_MT have the same absolute change. However, the execution rate of the component RPCM_MT is more sensitive in relative sense to the perturbations than FRITCS. The component RPCM_LT, which has the lowest stationary probability, has the highest value of the relative change of the execution rate. This implies that components with lowest stationary probability (i.e., the rarely executed components) are more sensitive in relative sense. This observation is very important for software verification and validation due to the fact that rarely executed components usually handle critical functionality such as for example exception handling or recovery. Further, note that the DTMS that describes the execution behavior of the HCS case study consists of a dominant central sate (FRITCS) with connections to and from all other states. For this type of software architectures, no component execution rate can be excessively sensitive to perturbations in the operational profile.

| States | S | PFMC_LT | PFMC_MT | FRITCS | RPCM_LT | RPCM_MT | SCITCS | E |
|---|---|---|---|---|---|---|---|---|
| **Stationary Probability** | 0.133 | 0.1 | 0.1 | **0.3** | **0.033** | **0.067** | 0.133 | 0.133 |
| **Absolute Change** | 0.289 | 0.333 | 0.333 | **0.4** | **0.367** | **0.4** | 0.289 | 0.289 |
| **Relative Change** | 2.167 | 3.333 | 3.333 | **1.333** | **11** | **6** | 2.167 | 2.167 |

**Table 3. Perturbation analysis of the HCS case study**

### 3.3. **E-commerce case study**

In this section we analyze the sensitivity of a typical e-commerce application. For illustration, we adopt the example of an e-commerce application presented in [Menasce02]. In the e-commerce applications the users interact with the Web sites through sessions that consist of consecutive request to execute e-business functions (search, add to cart, pay and so on) during a single visit to the site. In the example presented in [Menasce02], the user's navigation pattern within a session is captured by so called Customer Behavior Model Graph (CBMG). The CBMG describes how the users navigate through the site, which functions they use and the frequency of transitions form one function to the other function. Clearly, the CBMG corresponds to a discrete time Markov chain.

Due to a large number of diverse users, a single operational profile is not sufficient to describe the use of the Web site by different users. Thus, in [Menasce02] two operational profiles are given showing the usage of the same e-commerce site by two different types of users: an occasional buyer and a heavy buyer. DTMCs for these two operational profiles are shown in Figure 10 and Figure 11, respectively.

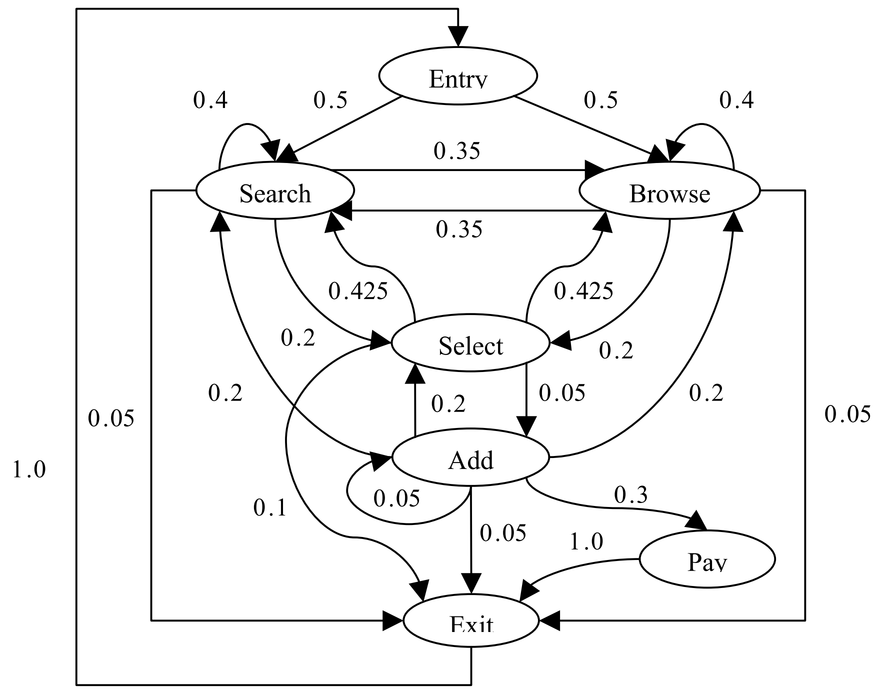


**Figure 10. Operational profile for an occasional buyer**

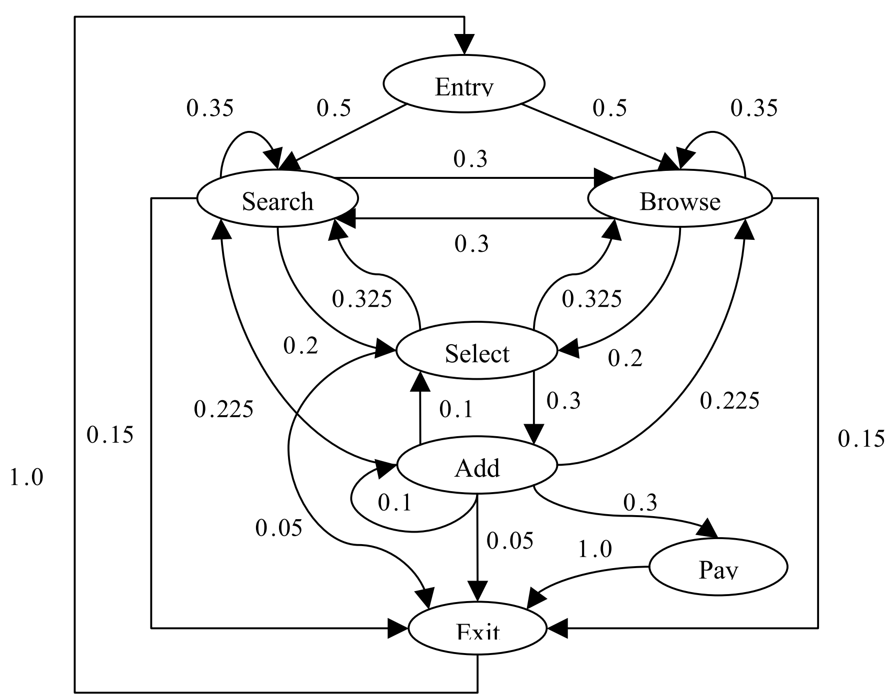


**Figure 11. Operational profile for a heavy buyer**

Construction of the DTMCs in Figures 10 and 11 includes constructing the structure first and then assigning transition probabilities. For Web applications, there is usually a close resemblance between navigation patterns and the underling Web design and code because Web sites are designed to support directly such navigations. Consequently, the basic structure can be easily identified from product specification, related design documents and other information sources, which corresponds to our intended approach. Also one might use the informed approach such as for example to extract the architecture from the HTML code or the Web access logs. Note that in our project *Performability of Web Based Applications,* funded by NASA OSMA Software Assurance Research Program (SARP) managed through the NASA Independent Verification and Validation Facility, we are addressing construction of the typical usage profiles from the Web access logs.

Here we assess the sensitivity of e-commerce application by perturbing the transition probability matrix P that represents the operational profile of an occasional buyer (i.e. DTMC shown in Figure 10). The matrix $\widetilde{P}$ is the resultant transition probability matrix, which represents the operational profile of a heavy buyer scenario (i.e. DTMC shown in Figure 11), after perturbing the matrix P by matrix E.

$$P = \begin{array}{c} \text{En} \\ \text{Br} \\ \text{Sr} \\ \text{Ad} \\ \text{Se} \\ \text{Pa} \\ \text{Ex} \end{array} \begin{bmatrix} 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0.4 & 0.35 & 0 & 0.2 & 0 & 0.05 \\ 0 & 0.35 & 0.4 & 0 & 0.2 & 0 & 0.05 \\ 0 & 0.2 & 0.2 & 0.05 & 0.2 & 0.3 & 0.05 \\ 0 & 0.425 & 0.425 & 0.05 & 0 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad E = \begin{array}{c} \text{En} \\ \text{Br} \\ \text{Sr} \\ \text{Ad} \\ \text{Se} \\ \text{Pa} \\ \text{Ex} \end{array} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.05 & -0.05 & 0 & 0 & 0 & 0.1 \\ 0 & -0.05 & -0.05 & 0 & 0 & 0 & 0.1 \\ 0 & 0.025 & 0.025 & 0.05 & -0.1 & 0 & 0 \\ 0 & -0.1 & -0.1 & 0.25 & 0 & 0 & -0.05 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\widetilde{P} = \begin{array}{c} \text{En} \\ \text{Br} \\ \text{Sr} \\ \text{Ad} \\ \text{Se} \\ \text{Pa} \\ \text{Ex} \end{array} \begin{bmatrix} 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0.35 & 0.3 & 0 & 0.2 & 0 & 0.15 \\ 0 & 0.3 & 0.35 & 0 & 0.2 & 0 & 0.15 \\ 0 & 0.225 & 0.225 & 0.1 & 0.1 & 0.3 & 0.05 \\ 0 & 0.325 & 0.325 & 0.3 & 0 & 0 & 0.05 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

As explained before the components with lowest execution rates are more sensitive relatively to changes in the operational profile. Thus, it can be observed from the results shown in Table 4 that the components '*Add*' and '*Pay*' which are visited rarely in the operational profile of an occasional buyer are having the highest relative changes, that is, they are the most sensitive in relative sense to the perturbation in P that leads to the operational profile of a heavy buyer. Thus, '*Add*' and '*Pay*' exhibit excessive relative change in execution rate due to the changes in the operational profile. In particular, the relative change of the expected execution rate of component '*Add*' is one order of magnitude higher and the relative change of the expected execution rate of component '*Pay*' is two orders of magnitude higher than execution rates of other components in the e-commerce case study.

| **States** | **Entry** | **Browse** | **Search** | **Add** | **Select** | **Pay** | **Exit** |
|---|---|---|---|---|---|---|---|
| **Stationary Probability** | 0.0542 | 0.3666 | 0.3666 | 0.0078 | 0.1482 | 0.0023 | 0.0542 |
| **Absolute Change** | 0.2365 | 0.4065 | 0.4065 | 0.2672 | 0.3149 | 0.2494 | 0.2365 |
| **Relative Change** | 4.3615 | 1.1089 | 1.1089 | 34.25 | 2.125 | 106.58 | 4.3615 |

**Table 4. Perturbation analysis of the e-commerce case study**

## 4. Comparison of the methods for uncertainty analysis

The choice of the most suitable method for uncertainty analysis for a particular application may be based on the following criteria:

- data requirements
- reliability measures derived
- accuracy of the solutions
- scalability with respect to the number of components.

In [Goseva02c] we have compared the entropy, method of moments, and Monte Carlo simulation accordingly to the above criteria. The characteristics of perturbation analysis as a method for uncertainty analysis are the following:

- *Data requirements*
  - Low: Point estimates of the transition probabilities.
- *Reliability measures derived*
  - NA: Reliability measures are not derived. Instead, we study the sensitivity of the expected execution rates of software components to perturbations in the operational profile.

- *Accuracy of the solutions*
    - Analytical solution; bounds for the absolute and relative change of components execution rates.
- *Scalability*
    - Scales well. Could be used for large systems.

Next we update our "Make a choice" table [Goseva02c], [Goseva02e] that summarizes the comparison of different methods for uncertainty analysis considered in FY02 and FY03. "Make a choice" table given in Table 5 provides a sound guideline for choosing the most appropriate method for a given software application accordingly to the above criteria.

| Method | Data requirements | Uncertainty of the operational profile | Reliability measures derived | Accuracy of the solution | Scalability |
|---|---|---|---|---|---|
| **Entropy** | Point estimates | ▪ Uncertainty of operational profile<br>▪ Uncertainty of components<br>▪ Expected execution rates of components | NA | Exact analytical solution | Large systems |
| **Method of moments** | Moments of components reliabilities | NA | Moments of system reliability | Approximate solution<br>▪ Accuracy may be increased by higher order Taylor series<br>▪ No sampling errors | Medium systems |
| **Monte Carlo simulation** | ▪ Probability distribution functions of components reliabilities and transition probabilities<br>▪ Generation of random numbers | NA | Many characteristics of system reliability<br>▪ Frequency chart<br>▪ Distribution<br>▪ Moments<br>▪ Percentiles | Approximate solution<br>▪Accuracy may be increased by increasing the sample size<br>▪Sampling errors may be involved in case of long tail distributions | Large systems |
| **Perturbation analysis** | ▪ Point estimates of transition probabilities | ▪ Bounds on the absolute and relative change of components execution rates | NA | Analytical solution | Large systems |

**Table 5. Make a choice table: Comparison of different methods for uncertainty analysis**

## 5. Conclusion

In this report we have presented a new method for uncertainty analysis based on perturbation theory that can be used to study how the change in the operational profile affects the expected execution rates of software components. We have applied this method on three case studies: European Space Agency, NASA's Hub Control System (HCS) from the International Space Station (ISS), and an e-commerce application. Several conclusions are made based on obtained results:

- If any stationary probability is relatively insensitive, then all large stationary probabilities are insensitive.
- If a small stationary probability is relatively insensitive, then it is absolutely insensitive and the operational profile is stable in both absolute and relative sense.
- Small stationary probabilities are the ones that appear least likely to be relative insensitive to perturbations.

Based on these conclusions we say that the stability of operational profile can be studied by looking at the small stationary probabilities. Further, components with small execution rates (i.e., small stationary probabilities) are the most sensitive to changes in the operational profile. This observation is very important for software verification and validation due to the fact that rarely executed components usually handle critical functionality such as for example exception handling or recovery.

The main focus of our future work is to develop the uncertainty method based on confidence intervals that can be used for certification of component based software systems. We will apply this method on the currently available case studies and any additional NASA cases studies that might be available in the future.

## 6. References

[Cho00] G. E. Cho and C. D. Meyer, "Markov Chain Sensitivity Measured by Mean First Passage Times", *Linear Algebra Appl.,* 316, 2000, pp. 21-28.

[Cho01] G. E. Cho and C. D. Meyer, "Comparison of Perturbation Bounds for the Stationary Distribution of a Markov Chain", *Linear Algebra Appl*., 335, 2001, pp.137-150.

[Goseva01a] K. Goseva-Popstojanova and K. S. Trivedi, "Architecture-Based Approach to Reliability Assessment of Software System", *Performance Evaluation*, Vol.45, No.2-3, 2001, pp.179-204.

[Goseva01b] K. Goseva-Popstojanova, A. P. Mathur, and K. S. Trivedi, "Comparison of Architecture-Based Software Reliability Models", *12th International Symposium on Software Reliability Engineering*, 2001, pp.22-31.

[Goseva02a] K. Goseva-Popstojanova and S. Kamavaram, "Architecture-Based Methodology for Studying Sensitivity of Software Reliability to Operational Profile Errors", *Technical Report,* March 2002.

[Goseva02b] K. Goseva-Popstojanova and S. Kamavaram, "Application of the Methodology for Uncertainty Analysis on Case Studies", *Technical Report,* September 2002.

[Goseva02c] K. Goseva-Popstojanova and S. Kamavaram, "Final Report", *Technical Report,* September 2002.

[Goseva02d] K. Goseva-Popstojanova and S. Kamavaram, "Uncertainty Analysis of Software Reliability Based on Method of Moments", *13th International Symposium on Software Reliability Engineering*, Nov 2002.

[Goseva02e] K. Goseva-Popstojanova and S. Kamavaram, "Assessing Uncertainty in Reliability of Component-Based Software", submitted for publication.

[Goseva02f] K. Goseva-Popstojanova and S. Kamavaram, "Uncertainty Analysis of the Operational Profile and Architecture-Based Software Reliability based on Entropy", submitted for publication.

[Hunter00] J. J. Hunter, "A Survey of Generalized Inverses and their Use in Stochastic Modeling", *Res. Lett. Inf. Math. Sci.,* 1, 2000, pp. 25-36.

[Hunter02a] J. J. Hunter, "Generalized Inverses, Stationary Distributions and Mean First Passage Times with Applications to Perturbed Markov Chains", *Res. Lett. Inf. Math. Sci.,* 3, 2002, pp. 99-116.

[Hunter02b] J. J. Hunter, "Stationary Distributions and Mean First Passage Times of Perturbed Markov Chains", *Research Letters in Information and Mathematical Sciences, Institute of Information and Mathematical Sciences, Massey University, Auckland, New Zealand,* 3, 2002 pp. 85-98.

[Ipsen94] I. C. F. Ipsen and C. D. Meyer, "Uniform Stability of Markov Chains", *SIAM J. Matrix Anal. Appl.,* 15, 1994, pp. 1061-1074.

[Kamavaram02] S. Kamavaram and K. Goseva-Popstojanova, "Entropy as a Measure of Uncertainty in Software Reliability", *13th International Symposium on Software Reliability Engineering*, Nov 2002.

[Menasce02] D. A. Menasce, "TCP-W A Banchmark for E-Commerce", *IEEE Internet Computing,* Vol.6, No.3, May/June 2002, pp.83-87.

[Meyer94] C. D. Meyer, "Sensitivity of the Stationary Distribution of a Markov Chain", *SIAM J. Matrix Anal. Appl.,* Vol 15, No. 3, July 1994, pp. 715-728.

[Schweitzer68] Paul J. Schweitzer, "Perturbation Theory and Finite Markov Chains", *J. Appl. Prob.* 5, 1968, pp. 401-413.